CLAIM LISTING

This listing of claims will replace all prior versions, and listings, of claims in the application.

1. (Currently Amended) A method for network packet processing comprises:

receiving network packets at a processor having multiple engines collectively providing multiple program threads, each of the multiple engines having multiple program counters for different program threads provided by the respective engine; and

operating on the network packets with a plurality of the program threads to affect processing of the packets, at least one of the threads accessing a first register that automatically resets to a null value after a read of the first register.

2. (Original) The method of claim 1 wherein operating comprises:

using at least one program thread to inspect a header portion of the packet.

3. (Original) The method of claim 2 wherein operating further comprises:

signaling by the at least one program thread that a packet header has been processed.

4. (Previously Amended) The method of claim 1, wherein the plurality of program threads comprise at least one scheduler program thread to schedule task orders for processing and processing program threads that process packets in accordance with task assignments assigned by the scheduler program threads.

5. (Currently Amended) The method of claim 1 wherein each program thread writes a message to a second register that indicates ~~its~~ the program thread's current status.

6. (Original) The method of claim 5 wherein interpretation of the message is fixed by a software convention determined between a scheduler program thread and processing program threads called by the scheduler program thread.

7. (Original) The method of claim 5 wherein status messages include busy, not busy, not busy but waiting.

8. (Original) The method of claim 5 wherein a status message includes not busy, but waiting and wherein the status of not busy, but waiting signals that the current program thread has completed processing of a portion of a packet and is expected to be assigned to perform a subsequent task on the packet when data is made available to continue processing of the program thread.

9. (Currently Amended) The method of claim 5 wherein the second register is a ~~globally accessible~~ register that can be read from or written to by different ones of the program threads executed by different ones of the multiple engines.

10. (Currently Amended) The method of claim 4 wherein the scheduler program thread ~~threads~~ can schedule ~~any~~ one of a plurality of processing program threads to handle processing of a task.

11. (Currently Amended) The method of claim 10 wherein the scheduler program thread writes a third register with an address corresponding to a location of data for the plurality of processing program threads.

12. (Currently Amended) The method of claim 11 wherein a ~~selected~~ one of the plurality of processing program threads that can handle the task reads the third register to obtain the location of the data.

13. (Currently Amended) The method of claim 12 wherein the ~~selected~~ one of the plurality of processing program threads reads the third register to obtain the location

of the data and to assign itself to processing the task requested by the scheduler program thread.

14. (Currently Amended) The method of claim 12 wherein the third register consists of the first register that automatically resets to a null value after a read of the first register, and wherein the ~~selected~~ one of the plurality of processing tasks reads the third register to obtain the location of the data~~, while the register is cleared by reading the register by the program thread to assign itself to process the task~~.

15. (Currently Amended) The method of claim ~~13~~ 14 wherein when another one of the plurality of processing program threads assignable to the task ~~attempts to read~~ reads the third register ~~after it has been cleared~~, it is provided with a null value that indicates that there is no task ~~currently assignable to~~ for the processing program thread.

16. (Currently Amended) A parallel hardware-based multithreaded processor for ~~receiving network packets~~ comprises:

~~a general purpose processor that coordinates system functions~~; and

a plurality of microengines that support multiple program threads, each of the multiple engines having multiple program counters for different program threads provided by the respective engine~~, and operate on the network packets with a plurality of program threads to affect processing of the packets~~;

and a first register that automatically resets to a null value after a read of the first register.

17. (Previously Amended) The processor of claim 16 wherein one of the plurality of microengines executes at least one scheduler program thread and other ones of the microengines execute processing program threads.

18. (Currently Amended) The processor of claim 16 further comprising a ~~global thread status~~ second register wherein ~~each~~ multiple program threads write ~~program~~

~~thread writes~~ a message to the ~~global~~ second status register that indicates ~~its~~ the threads respective current status.

19. (Original) The processor of claim 18 wherein interpretation of the message is fixed by a software convention determined between a scheduler program thread and processing program threads called by the scheduler program thread.

20. (Currently Amended) The processor of claim 16 ~~further comprising:~~

~~a read once register~~, wherein ~~the scheduler~~ a program thread writes the ~~read once~~ first register ~~with an address corresponding to a location of data for the plurality of processing program threads~~ and when a ~~selected~~ one of the plurality of processing program threads reads the register ~~to obtain the location of the data~~, the one of the plurality of processing program threads assigns itself to processing ~~the~~ a task ~~requested by the scheduler program thread, while the register is cleared by reading the register by the program thread~~.

21. (Currently Amended) The processor of claim 20 wherein when another one of the plurality of processing program threads ~~assignable to the task attempts to read the read once~~ reads the register ~~after it has been cleared~~, ~~it~~ the another one of the plurality of processing program threads is provided with a null value that indicates that there is no task to assign ~~currently assignable to the processing program thread~~.

22. (Currently Amended) An apparatus comprising a machine-readable storage medium having executable instructions for network processing, the instructions enabling the apparatus to:

receive network packets; and

operate on the network packets with a plurality of program threads collectively provided by multiple engines of a processor to affect processing of the packets, each of the multiple engines having multiple program counters for different program threads provided by the respective engine.

wherein at least some of the plurality of program threads access a first register to determine if a packet processing task is awaiting performance, and

wherein if a one of the plurality of program threads accesses a non-null first register value representing a packet processing task awaiting performance, the first register is reset to a null value and the one of the plurality of program threads causes the packet processing task to be performed.

23. (Original) The apparatus of claim 22 wherein instructions to operate further comprise instructions to:

use at least one program thread to inspect a header portion of the packet.

24. (Original) The apparatus of claim 22 further comprising instructions to provide scheduler program threads to schedule task orders for processing and processing program threads to process packets in accordance with task assignments assigned by the scheduler program threads.

25. (Currently Amended) The apparatus of claim 22 wherein each program thread writes a message to a second register that indicates ~~its~~ the thread's current status.

26. (Currently Amended) The apparatus of claim 25 wherein the register is a ~~globally~~ accessible register that can be read from or written to by all current program threads.

27. (Currently Amended) The apparatus of claim 22 wherein the first register comprises a first register that automatically resets to a null value upon a read of the first register ~~wherein the scheduler program thread writes a register with an address corresponding to a location of data for the plurality of processing program threads and a selected one of the plurality of processing program threads that can handle the task reads the register to obtain the location of the data, and clears the register after reading by the program thread~~.

28. (Original) The apparatus of claim 27 wherein when another one of the plurality of processing program threads ~~assignable to the task attempts to read~~ reads the first register ~~after it has been cleared~~, ~~it~~ the one of the plurality of processing program threads is provided with a null value that indicates that there is no task ~~currently assignable to~~ for the processing program thread.

29-35. (Withdrawn)

36. (Cancelled)

37-44. (Withdrawn).